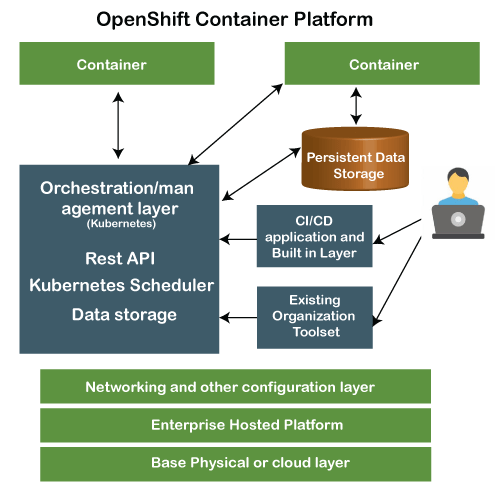
**OpenShift:**

* PaaS, cloud development platform.
* extensible (helps users to write code in other language).
* helps to move from traditional application/platform to the cloud.
* variety of application, quickly developed and deployed on the OpenShift. 3 kinds of platforms – IaaS, PaaS, SaaS.
* A layered system each layer is tightly bound with other layer using Kubernetes and Docker cluster.
* Design to support and manage Docker containers, hosted on the top of all the layers.
* Supports containerized infrastructure. Docker creates lightweight Linux-based containers, Kubernetes supports the task of orchestrating and managing containers on multiple hosts.
* Key difference from Kubernetes is the concept of build-related artifacts.
* In OpenShift, such artifacts are considered first-class Kubernetes resources upon which standard Kubernetes operations can apply. OpenShift's client program, known as oc, offers a superset of the standard capabilities bundled in the mainline kubectl client program of Kubernetes. Using this client, one can directly interact with the build-related resources using sub-commands.
* an OpenShift-native pod build technology called Source-to-Image (S2I) is available out of the box.
* Common platform for enterprise units to host their applications on the cloud without worrying about the underlying OS. This makes it easy to use develop and deploy the application on cloud.
* Managed hardware and network resources for development and testing. PaaS has the freedom to design their required environment with specifications.
* 3 types of service plans: Free, Bronze, Silver.
* Components: Cartridges, Gear, Application.
* different versions: OpenShift Origin, OpenShift Online, OpenShift Enterprise, OpenShift Container Platform, OpenShift Container Lab, OpenShift Dedicated.
* OpenShift container Platform: enterprise platform- helps multiple teams to build and deploy containerized infrastructure. All the containers built-in OpenShift uses a very reliable Docker containerization technology, which can be deployed on any data center of publicly hosted cloud platform.
* Brings Docker and Kubernetes together to the enterprise level.
* 2 package level: OpenShift Container Local, OpenShift Container Lab.

|  |  |  |
| --- | --- | --- |
| OpenShift Versions | Key features |  |
| OpenShift Origin | * community addition or open-source version * known as an upstream project for the other two versions. |  |
| OpenShift Online | * public PaaS as a service hosted on AWS. * build, deploy, and scale containerized applications on public cloud. * automated provisioning, management, and scaling of application, which helps the developer focus on writing application logic. |  |
| OpenShift Enterprise | * hardened version of OpenShift with ISV and vendor licenses. * an enterprise platform which helps multiple teams such as development and IT operations team to build and deploy containerized infrastructure. * Formally known as OpenShift Container platform * a Red Hat on-premises private platform as service, concept of application containers powered by Docker, where Kubernetes manage orchestration and administration. * two package levels: OpenShift Container Local, OpenShift Container Lab. | Features:   * Extensible and Open * Portability * Orchestration * Automation |
| OpenShift Community Distribution | * upstream community project used in OpenShift Online, OpenShift Dedicated, and OpenShift Container Platform * Origin is augmented by application lifecycle management functionality and DevOps tooling. * Origin provides an open-source application container platform. All source code for the Origin project is available under the Apache License on GitHub. |  |
| OpenShift Dedicated | * customer choice of hosting a containerized platform on any of the public cloud of their choice. * Gives end-user a true sense of multi-cloud offering, they can use OpenShift on any cloud which satisfies their needs. * Used for building, testing, deploying, and running their application on OpenShift which is hosted on the cloud. |  |

**Global, regional, and zonal resources:**

* Google Cloud resources are hosted in multiple locations worldwide. These locations are composed of regions with zones within those regions. Putting resources in different zones in a region provides isolation from many types of infrastructure, hardware, and software failures. Putting resources in different regions provides an even higher degree of failure independence. You can design robust systems by spreading resources across different failure domains.
* All Compute Engine resources are either global, regional, or zonal. For example, images are a global resource, but persistent disks are either regional or zonal resources. The scope of the resource determines how accessible the resource is to other resources. For example, global resources are accessible by resources in any region or zone, so virtual machine (VM) instances from different zones can use the same global image.
* Regional resources are accessible only to resources within the same region. For example, a regional static external IP address is accessible only by resources within the same region.
* Resources that are hosted in a zone are called per-zone resources. Zone-specific resources, or per-zone resources, are unique to that zone and are only usable by other resources in the same zone. For example, an instance is a per-zone resource. When you create an instance, you must provide the zone where the instance is located. Resources that line in a zone like virtual machine instances are known as zonal resources, whereas other resources like static external IP addresses are regional and can be sued by any resource within the region regardless of the zone. Zonal resources can only be used by other resources within the same zone.

**Cabling:**

* Connecting various components within a datacenter.
* Essentially defined as the networking of cables which are used in a typical data center.
* Two types: Structured and unstructured.
* Structured Cabling: predefined cabling standard, pre-set connection points and pathways. tests to ensure high performance along with appropriate organization and labelling of cables. the overall operation costs are lower, and the lifespan of the entire system is longer.
* Unstructured Cabling: not use any predefined standards, connection points or pathways, which may result in higher energy costs and difficult cable management.
* Types:
  + AC/DC power cables:
  + Ground:
  + Copper:
  + Fiber Optic cables:

**East-West and North-South network traffic flow:**

